Qcd2

Min^m cost Climbing stairs

by recursion

$$f(n) \Rightarrow f(n+1) + f(n+2)$$

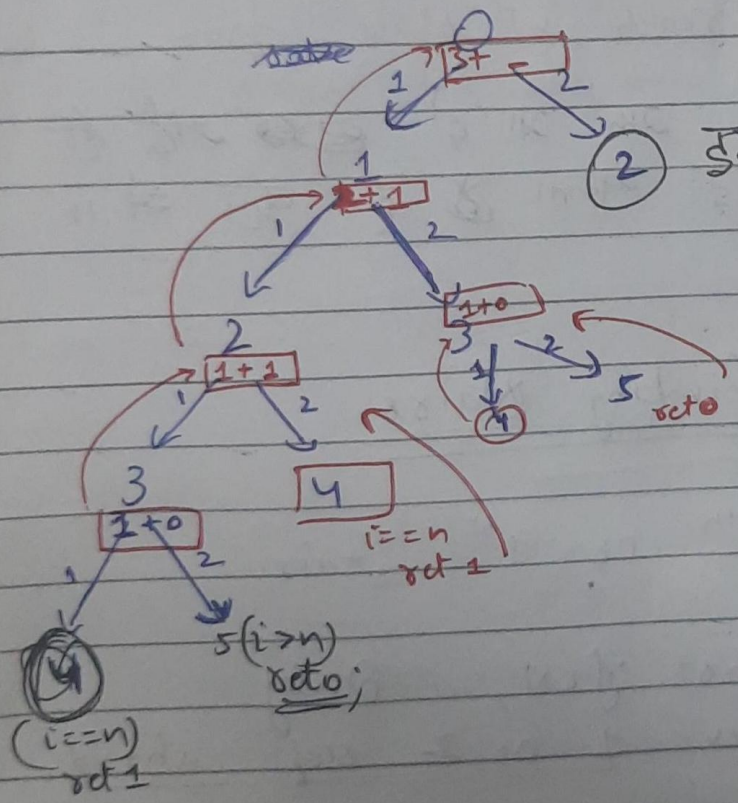
or

$$f(n) \Rightarrow f(n-1) + f(n-2)$$

if (n==i) ret 1;
if (n<i) ret 0;
ret Solve(n,i+1)+
solve(n,i+2);

dry run

n=4



इसका answer तो मैं पहले ही निकाल चुका हूँ ; तो मुझे yir DP लगेगी ||

" Overlapping sub-problems "

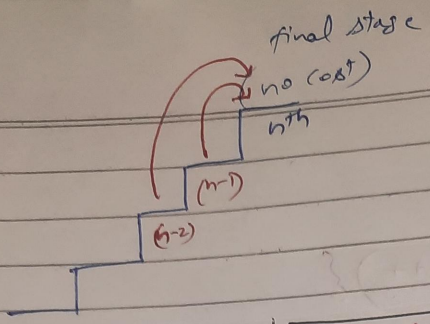
If we write f(k) ; then
it means function should total possible way to go from k^{th} stair to n^{th} stair by 1 or 2 step.

If need to find no. of way to go from 0^{th} stair n^{th} stair
तो मैं इस problem को एक subproblem में तोड़ सकता हूँ ॥

⇒ To can say, it will be equal to no of way to reach n^{th} stair from 1^{st} stair plus ways to reach n^{th} stair from 2^{nd} stair. (क्योंकि एक या दो ही चढ़ सकता हूँ यहाँ से ॥

leetcode : Min Cost Climbing Stairs

1. Can start from 0^{th} or 1^{st} stair
2. have to pay the cost first, Only then we can move 1 or 2 step ahead.
3. Return min^n cost to reach last Top stage



$$f(n) \Rightarrow \text{cost}(n) + \min\left(f(n-1) + f(n-2)\right)$$

Recursion

```
int solve (vector<int>& cost, int n){
    if (n==0)
        return cost[0];
    if (n==1)
        return cost[1];

    int ans = cost[n] + min(solve(cost,n-1) + solve(cost,n-2));

    return ans;}

int minCost (vector<int>cost){

int ans = min(solve(cost,n-1), solve(cost,n-2));

}
```

Bottom up (dp)

wow

```
int Solve3 (vector<int>& cost, int n) {
    vector<int> dp(n+1); //step1.
```

```
dp[0] = cost[0];
dp[1] = cost[1];

for (i=2; i<n; i++){
    dp[i] = cost[i] + min(dp[n-1], dp[n-2]);
}

return min(dp[n-1], dp[n-2]);

}
```

Space optimization भी हो जाएगा !!

हर value; अपने previous last two values पर depend करती है ॥